Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

Bayesian Clustering and Tracking of Neuronal Signals for Autonomous Neural Interfaces

Michael T. Wolf and Joel W. Burdick

Abstract— **This paper introduces a new, unsupervised method for sorting and tracking the non-stationary spike signals of individual neurons in multi-unit extracellular recordings. While this method may be applied to a variety of problems that arise in the field of neural interfaces, its development is motivated by a new class of autonomous neural recording devices. The core of the proposed strategy relies upon an extension of a traditional expectation–maximization (EM) mixture model optimization to incorporate clustering results from the preceding recording interval in a Bayesian manner. Explicit filtering equations for the case of a Gaussian mixture are derived. Techniques using prior data to seed the EM iterations and to select the appropriate model class are also developed. As a natural byproduct of the sorting method, current and prior signal clusters can be matched over time in order to track persisting neurons. Applications of this signal classification method to recordings from macaque parietal cortex show that it provides significantly more consistent clustering and tracking results than traditional methods.**

I. Introduction

The need to reliably identify and track the activities of a particular neuron in multi-unit extracellular recordings is a common problem in basic electrophysiological studies and engineered neural interfaces. Much of what we know about brain function has been provided by extracellular neural recordings, which are obtained by positioning the tip of an electrode near a neuron to detect and localize in time the occurrence of the neuron's *action potentials* or *spikes*, which are the basis for neural communication and information processing. The electrode tip may happen to be within the "listening sphere" of multiple neurons, however, causing the spiking activity of several neurons to be recorded on a single electrode. In general, the interpretation of all extracellular recordings requires a process to associate the experimental data with the activity of individual neurons over the duration of the recording, commonly referred to as "spike sorting" (see [1] for a review). We address here the challenge of autonomously classifying these spikes according to their generating neuron and tracking the identities of the neurons over time, even as their signal characteristics may change.

While this problem is similar in many respects to conventional multi-target tracking problems, there are significant differences that motivate our work. As described below, the measurement and identification processes involve the sequential clustering of noisy data over time; in effect, the neuron "target" may be observed only through the population of spikes it generates during every sampling interval. Since the success of the tracking solution in this case is highly dependent upon the quality and consistency of the clustering process, the work in this paper may be interpreted as a clustering-based version of tracking and data association filters.

Although our solution may be applied to a variety of applications, it is particularly motivated by related work in which the authors have developed a set of control algorithms (and a novel miniature "robotic" multi-electrode recording device) that autonomously move electrodes to find individual neurons, optimize their signal quality, and then maintain the signal over time in the presence of tissue migration and mechanical shocks [2]–[5]. This algorithm implements a form of extremum-seeking control. In the control algorithm's main loop, the electrode's signal is sampled for an interval of, say, $T = 10$ seconds. The neuronal spikes must then be identified from this voltage trace sample and sorted according to their generating neurons. The algorithm selects the neuron whose signals have the highest average signal-to-noise ratio (SNR)—the "dominant" neuron—and determines what electrode movement, if any, is needed to increase signal quality. The performance of this algorithm crucially depends upon accurate spike clustering and neuron tracking, as incorrect clustering will corrupt the SNR observations and effective data association is required to monitor a neuron's SNR over time. Moreover, the amplitude, phase, and numbers of neuronal signals will vary due to the electrode's movement, complicating the challenge.

The process of sampling data across sequential intervals is a key distinguishing element of our problem statement, and it also arises in other neural interface applications. For example, during the training phase of many brain-machine interfaces, signals from multi-electrode arrays implanted in cortex are sampled during repetitive execution of a task, which typically lasts a relatively short duration (e.g. $T =\sim 5$ seconds). In order to properly estimate the tuning properties of the neurons sampled by the array, the signal sources must be sorted on each electrode and matched, or tracked, across each repetition of the task execution. More generally, in multi-unit recordings gathered during basic scientific studies, it can be useful to divide lengthy recordings into short time windows for spike sorting and analysis, as the data are apt to be effectively stationary over these intervals but non-stationary over longer periods due to electrode drift (and other causes). Here again, the resulting neurons must be matched across analysis intervals. Often these tasks are

This work was completed at the California Institute of Technology with support from the National Institutes of Health and the Rose Hills Foundation.

M. T. Wolf is now with the Jet Propulsion Laboratory, Pasadena, CA. wolf@jpl.nasa.gov

J. W. Burdick is with the California Institute of Technology, Pasadena, CA. jwb@robotics.caltech.edu

achieved manually, but because of our motivating interest in autonomous microdrives and neural interfaces, we require *unsupervised* methods applicable to single-channel electrodes in *online* recording — or at least in small, real-time batches.

Because of its importance and difficulty, unsupervised spike sorting has received great attention. Many traditional clustering procedures have been adapted to classify neural waveforms, including hierarchical [6], k-means [7], neural networks [8], superparamagnetic [9], and template matching [10]. The optimization of a (typically Gaussian) mixture model has been shown to be a particularly effective, and often superior, approach [1], [11], [12]. However, most of these techniques are designed for offline processing of large data sets. When the recording process requires repeated sampling and clustering over time, our experience has shown that the inconsistency of the output of conventional clustering methods prevents accurate tracking of the neurons' identities across sampling intervals.

To increase the consistency of clustering results over short successive time windows, we have created a model-based clustering technique that incorporates the available information over time, using the clustering results of the data set sampled during interval $k-1$ to improve the clustering of the subsequent set of data sampled during interval k, etc. Our proposed approach can be interpreted as a data association and tracking filter that operates on processes that generate clusters of measurements. This procedure is designed to succeed even in low firing rates (few samples per cluster), low signal-to-noise ratio, poor cluster separability, and non-stationary waveforms. Bar-Hillel et al. [12] are, to our knowledge, the only others to include neighboring clustering results for such a purpose, and the only others to address the novel problem of joint clustering and tracking procedures, but present a non-causal, computationally-intensive method designed for offline processing and hence not applicable to the real-time applications that motivate our work.

Section II reviews a classical maximum likelihood (ML) clustering method based on expectation–maximization (EM) over a Gaussian Mixture model, so that our subsequent extensions to this method can be more clearly delineated. Section III details our proposed method for sequential clustering based on Bayesian parameter estimation and model selection, while Section IV discusses how neurons are tracked based on the output of the clustering process. Applications of this method to neural recordings in macaque parietal cortex are presented in Section V and discussed in Section VI, where we provide characterizations of our method and comparisons to other clustering techniques.

II. ML Optimization of Mixture Models via EM

The spike sorting task has traditionally been posed as a clustering problem, due to the poor SNR of recorded neural signals and the variability in the shapes of the spike waveforms. While it is strongly believed that the timing of the spikes, and not their waveform shapes, carries information, the recorded spike waveforms of disparate neurons differ sufficiently to allow separation based on waveform characteristics. To prepare for clustering, first all action potential waveforms (possibly from different neurons) are detected and extracted from the sampling interval. Next, these waveforms are projected onto a d-dimensional feature space in which each spike is represented as a point.

The classical clustering technique of maximum likelihood (ML) optimization of a mixture model [13], [14] has been the basis for several spike sorting algorithms [1]. The underlying assumption of the mixture model approach is that each mixture component represents a neuron, which produces samples (spike features) according to a probability distribution. For example, if the ith spike waveform (in feature space) $y_i \quad \mathbb{R}^d$ was generated by the gth neuron (belongs to component, or cluster, $\mathcal{C}_g$), then it is governed by the probability density $p(y_i \;\; i \quad \mathcal{C}_g, \theta_g) = f_g(y_i \;\; \theta_g)$. If we assume a Gaussian (normal) PDF, then the parameters of the distribution are its mean and covariance matrix, i.e., $\theta_g = \;\; \mu_g, \Sigma_g \;\;$.

Including all N data points in the recording interval and all mixture components $g = 1, ..., G_m$, the *mixture likelihood* is

$$\mathcal{L}_M(\Theta_m) = p(Y \;\; \Theta_m, \mathcal{M}_m) = \prod_{i=1}^{N} \sum_{g=1}^{G_m} \pi_g f_g(y_i \;\; \theta_g), \quad (1)$$

where $Y = \;\; y_i \;\; {}_{i=1}^{N}$ is the set of all observations; $\mathcal{M}_m$ is the mth model class under consideration in the current recording interval, which dictates the model order G_m (i.e., the number of individual neurons contributing to the extracellular signal), the form of the gth probability density f_g (typically Gaussian), and the form of the model parameters Θ_m, which include θ_g and π_g; and π_g is the mixture weight of component $\mathcal{C}_g$ (the prior probability that an observed spike belongs to component $\mathcal{C}_g$).

The EM algorithm [15] is typically applied to estimate the mixture parameters by log-likelihood maximization, To apply this technique, we view our data Y as "incomplete" and augment it by Z, the set of membership variables $z_i = (z_{i1}, ..., z_{iG_m})$,

$$z_{ig} = \begin{array}{ll} 1 & \text{if spike waveform } y_i \text{ belongs to cluster } \mathcal{C}_g \\ 0 & \text{otherwise} \end{array} .$$

Incorporating Z one can derive the corresponding *complete-data log-likelihood*

$$p(Y, Z \;\; \Theta_m, \mathcal{M}_m) = \prod_{i=1}^{N} \sum_{g=1}^{G_m} z_{ig} \log \left[\pi_g f_g(y_i \;\; \theta_g)\right] . \quad (2)$$

EM iterates between an E-step to calculate the conditional expectation $\hat{z}_{ig} = E[z_{ig} \quad y_i, \hat{\Theta}_m] \quad [0, 1]$ using the current parameter estimates, and an M-step to find the parameter estimates $\hat{\Theta}_m$ that maximize (2) given $\hat{z}_{ig}$. This iteration guarantees (under weak conditions) a non-decreasing $\mathcal{L}_M$ (1) and is continued until a convergence threshold. Note that EM implicitly assigns data points to the appropriate mixture component via $\hat{z}_{ig}$, thereby effecting the clustering of the spike waveforms. However, this approach uses only data

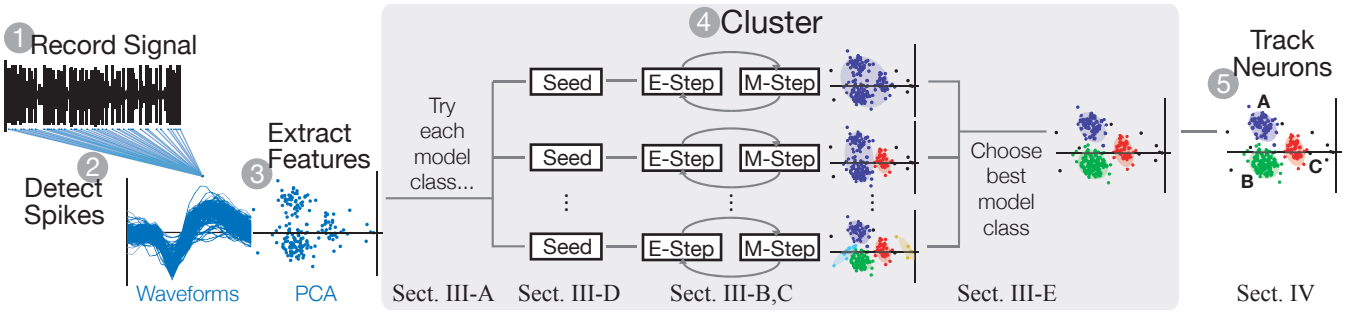

Fig. 1. Structure of the algorithm. Before clustering takes place, 1) the electrode signal is recorded for a brief sampling interval; 2) neuronal spike waveforms are detected in this voltage trace and aligned by their minimum; and 3) spikes are projected onto an appropriate feature space, such as a 2-dimensional PCA basis. 4) Next, these data points are clustered using EM mixture model optimization, over several possible model classes. 5) Finally, neurons are tracked by associating the clusters from the current interval to the previous interval. This entire process is repeated for every sampling interval.

sampled during a single interval, and does not incorporate any data or clustering results from prior sampling intervals.

III. MAP Clustering for Neuron Tracking

Fig. 1 outlines the general flow of the spike sorting process, with particular attention to the clustering step. Recall that some preprocessing steps are required to detect and extract spikes and then project them to a feature space. We use a 2-dimensional principal component (PCA) basis, a common convention in the neuroscience community, although our technique may be applied in any feature space. In the clustering block, EM iterations must be initialized by "seed clusters," or an initial guess (see Sect. III-D). Also, the EM algorithm assumes that the model class — most importantly, the number of clusters G_m — is assumed known *a priori*, but this is not feasible for spike sorting and many other applications. We employ a typical workaround of running EM for several different model classes $\mathcal{M}_m$, $m = 1, ..., \bar{M}$, varying G_m among them, and then evaluating the results of each model to select the best.

Our primary technical innovations in this procedure lie in four parts. First, we convert the EM algorithm to MAP optimization (rather than ML) of a mixture model to enable improved clustering and tracking over sequential intervals. Although MAP optimization via EM has previously been proposed for generic clustering cases (for example [14]), we explicitly derive a mixture prior appropriate to our application and the resulting EM adjustments. Second, our method uses the prior clusters to provide appropriate seed clusters, thereby increasing the chances of avoiding poor local optima in the EM process. Similarly, the model selection procedure incorporates information from the preceding results while still admitting changes in the number of recorded neurons. Finally, this solution inherently provides a simple tracking method to estimate whether clusters in consecutive recording intervals can be associated to the same neuron.

Let us now incorporate the sequential nature of the data sampling process, establishing the Bayesian framework for MAP parameter estimation (determining model parameter estimates $\hat{\Theta}_m$ and thus cluster membership Z) and model selection (determining the most appropriate number of clusters, $\hat{G}$). Let $Y^k = \{y_i\}_{i=1}^N$ be all spike samples in the kth sampling interval and $Y^{1:k} = \{Y^1, ..., Y^k\}$ denote data from the 1st through the kth intervals. The MAP parameter estimates can be naturally derived from Bayes' Rule:

$$p(\Theta_m^k \mid Y^{1:k}, \mathcal{M}_m) \propto \underbrace{p(Y^k \mid \Theta_m^k, \mathcal{M}_m)}_{\text{likelihood, Eq. (1)}} \underbrace{p(\Theta_m^k \mid Y^{1:k-1}, \mathcal{M}_m)}_{\text{prior, Sect. III-B}}$$

where we have removed the unnecessary conditioning on $Y^{1:k-1}$ in the likelihood, and Θ_m^k denotes the mixture model parameters of the mth model of the kth sampling interval.

A. Model Classes

While many model classes are possible within the framework used in this paper, we focus here on model classes that yielded the best results for neuronal signals in a PCA basis. Since the number of neurons contributing to a recorded signal is not known *a priori* and can change during the recording session, the model classes must allow for the presence of different possible numbers of neurons, $G_m = 1, ..., G_{\max}$, in the recorded signal (we typically use $G_{\max} = 5$). We choose a Gaussian distribution, whose PDF is denoted $f_{\mathcal{N}}$, to account for the variability in each neuron's signals. We also use a parsimonious shared-volume model of the covariance matrices Σ_g (see [16]), finding it often provides better results than the traditional fully variable model, but this is not a requirement of our approach. Commonly, non-spike events are included in the data Y^k and must be identified as *outliers*. A uniform "background" distribution $f_0 = \frac{1}{V}$ is added to the mixture model in order to capture these points, where V is the volume of the measurement space. Thus, the mixture likelihood (1) can be rewritten as a sum from $g = 0, ..., G_m$ where the zeroth component is the outlier distribution. The set of independent parameters for the Gaussian mixture model is $\Theta_m^k = \{\mu_g^k, \Sigma_g^k, \pi_g^k\}_{g=1}^{G_m}$.[1]

B. Prior on Cluster Location

Next, we construct an appropriate prior on the model parameters based on the previous sampling interval's clusters.

[1]The parameter set includes only the independent elements of the (symmetric) Σ_g^k, which will depend on the chosen covariance model. We will treat the matrix as a single parameter for brevity. Also note that $\pi_0^k = 1 - \sum_{g=1}^{G_m} \pi_g^k$.

The model parameters are assumed to be independent across mixture components and across each parameter; therefore,

$$p\big(\Theta_m^k \;\; Y^{1:k-1}, \mathcal{M}_m\big) = \prod_{g=1}^{G_m} p\big(\mu_g^k \;\cdot\big) p\big(\Sigma_g^k \;\cdot\big) p\big(\pi_g^k \;\cdot\big)\,,$$

where the factors on the right are the prior probability densities of the respective mixture model parameters with the same conditioning as on the left hand side. Most important to the practical issue of cluster consistency and tracking is the location of each cluster center, μ_g. Since the cluster covariance Σ_g and the mixture weight π_g associated with a given neuron may vary substantially from one sampling interval to the next, the previous values for these parameters may not be informative, and thus we choose diffuse priors for these model elements.

To establish priors on the cluster center locations, we look for the gth cluster mean μ_g to be near to *any* of the preceding interval's cluster locations, without regard to which one, and thus utilize mixture of Gaussians representing all of the previous interval's cluster means. To allow for the possibility that $\mathcal{C}_g$ represents a *new* neuron that was not recorded in the previous interval, a uniform distribution component is included as well. Thus, the prior on each mean is

$$p\big(\mu_g^k \;\; \cdot\big) \;=\; \frac{\omega_0}{V} + \sum_{j=1}^{\hat{G}^{k-1}} \omega_j^k f_{\mathcal{N}}\big(\mu_g^k \;\; \hat{\mu}_j^{k-1}, S_j^{k\;k-1}\big)\,, \quad (3)$$

where the first term represents the uniform component and the sum consists of the Gaussian components of all $\hat{G}^{k-1}$ clusters estimated in the previous interval. The parameter $\hat{\mu}_j^{k-1}$ is the estimated value of the jth cluster mean in the previous interval, and $S_j^{k\;k-1}$ is the covariance associated with the estimation that the current mean μ_g^k is in the same location as the prior mean $\hat{\mu}_j^{k-1}$. To set this parameter, $S_j^{k\;k-1} = R_j^{k-1} + Q^{k-1}$, where $R_j^{k-1} = \frac{1}{N_j^{k-1}}\Sigma_j^{k-1}$ is the measurement covariance matrix associated with our estimation of $\hat{\mu}_j^{k-1}$ (N_j is the number of data points in cluster $\mathcal{C}_j$) and the process noise covariance matrix Q^{k-1} is determined empirically. The value for Q^{k-1} will depend on the recording interval length T, the time between recording intervals (if any), and experimental setup (e.g., amplification values), and can be adjusted to account for possible electrode movement or other temporal factors. The mixture weight ω_j, which is the prior probability of assigning a cluster to the jth neuron, can be weighted by the prior probabilities that the jth neuron will be detected on the current sampling interval, $\mathcal{P}_{\mathrm{d},j}$, or that a new neuron will be detected, $\mathcal{P}_{\mathrm{n}}$:

$$\omega_j = \begin{cases} \frac{\mathcal{P}_{\mathrm{n}}}{c} & j = 0 \\ \frac{\mathcal{P}_{\mathrm{d},j}}{c} & j = 1, ..., \hat{G}^{k-1} \end{cases}$$

where c is the normalization constant. While this formulation allows the probability of detection to vary among known neurons (perhaps according to their firing rates), the results presented in this paper use a common $\mathcal{P}_{\mathrm{d}}$.

C. Extending EM to Account for Cluster Location Priors

Note that the prior (3) bears distinct resemblance to the mixture likelihood (1) and will in fact share the same difficulty of maximization. The same solution approach can be used: add hidden variables and optimize via EM. Thus new "membership" variables, $\mathcal{Z}^k = \zeta_{gj}^k$, are added to indicate whether the previously found (time $k-1$) cluster $\mathcal{C}_j$ should influence the current (time k) cluster $\mathcal{C}_g$, or, ideally,

$$\zeta_{gj}^k = \begin{cases} 1 & \text{if } \mu_g^k \text{ and } \hat{\mu}_j^{k-1} \text{ represent the same neuron} \\ 0 & \text{otherwise.} \end{cases}$$

Based on this approach, instead of using (3) directly, we employ the following *complete-data* log prior on the means:

$$\log p\big(\mu^k, \mathcal{Z}^k \;\; \hat{\Theta}^{k-1}\big) = \sum_{g=1}^{G_m} \sum_{j=0}^{\hat{G}^{k-1}} \zeta_{gj}^k \log\left[\omega_j^k f_j\big(\mu_g^k \;\; \psi_j^k\big)\right] \quad (4)$$

where ψ_j^k are the parameters of jth mixture component in the prior ($\psi_j^k = \hat{\mu}_j^{k-1}, S_j^{k\;k-1}$ for the Gaussian components).

Rewriting Bayes' Rule to include the hidden variables and taking the log results in

$$\log p\big(\Theta_m^k, \mathcal{Z}^k \;\; Y^{1:k}, Z, \mathcal{M}_m\big) = \mathcal{C} + \underbrace{\log p\big(Y^k, Z^k \;\; \Theta_m^k, \mathcal{Z}^k, \mathcal{M}_m\big)}_{\text{Eq. (2)}} + \underbrace{\log p\big(\Theta_m^k, \mathcal{Z}^k \;\; Y^{1:k-1}, \mathcal{M}_m\big)}_{\text{Eq. (4)}}$$

where $\mathcal{C}$ is a constant. This *complete-data posterior* is the object equation of the EM algorithm's iterations, which follow.

1) E-Step: As in the classical EM algorithm, the E-step calculates the expectation of the spike-to-cluster memberships $\hat{z}_{ig}$, given the parameter estimates $\hat{\Theta}_m^k$:

$$\hat{z}_{ig}^k = \frac{\hat{\pi}_g^k f_g\big(y_i \;\; \hat{\theta}_g^k\big)}{\sum_{n=0}^{G_m} \hat{\pi}_n^k f_n\big(y_i \;\; \hat{\theta}_n^k\big)}\,.$$

The expectation of the other hidden data, the current-to-prior cluster membership $\hat{\zeta}_{gj}^k = E\;\; \zeta_{gj}^k \;\; Y^{1:k}, \hat{\Theta}_m^k\;$, is:

$$\hat{\zeta}_{gj}^k = \frac{\omega_j^k f_j\big(\hat{\mu}_g^k \;\; \psi_j^k\big)}{\sum_{l=0}^{\hat{G}^{k-1}} \omega_l^k f_l\big(\hat{\mu}_g^k \;\; \psi_l^k\big)}\,. \quad (5)$$

2) M-Step: Since the prior is independent of the parameters π_g and Σ_g, these estimates remain the same as the classical ML clustering version [13]. Maximizing our object equation with respect to μ_g results in the estimate:

$$\hat{\mu}_g^k = \left[\sum_{i=1}^{N} \hat{z}_{ig}^k (\hat{\Sigma}_g^k)^{-1} + \sum_{j=1}^{\hat{G}^{k-1}} \hat{\zeta}_{gj}^k (S_j^{k\;k-1})^{-1}\right]^{-1} \cdot \left[\sum_{i=1}^{N} \hat{z}_{ig}^k (\hat{\Sigma}_g^k)^{-1} y_i + \sum_{j=1}^{\hat{G}^{k-1}} \hat{\zeta}_{gj}^k (S_j^{k\;k-1})^{-1} \hat{\mu}_j^{k-1}\right], \quad (6)$$

in contrast to the ML estimation of the cluster center location,

$$\hat{\mu}_g^k = \frac{\sum_{i=1}^{N} \hat{z}_{ig}^k y_i}{\sum_{i=1}^{N} \hat{z}_{ig}^k}.$$

Note that Equation (6) has the form of a weighted average of the data points y_i with (fuzzy) membership to cluster $\mathcal{C}_g$ and the prior means $\hat{\mu}_j^{k-1}$ (fuzzily) affiliated to cluster $\mathcal{C}_g$, with the weights governed by the respective covariance matrices. A minor drawback to the MAP parameter calculation is that (6) is a function of the parameters $\hat{\Sigma}_g^k$, implying the need to simultaneously solve the equations for the parameters $\hat{\mu}_g^k$ and $\hat{\Sigma}_g^k$. Alternatively, one may use an approximation for $\hat{\Sigma}_g^k$ to solve (6), such as its value from the previous EM iteration, and then find $\hat{\Sigma}_g^k$ in the usual way.

D. Generating Seed Clusters

While the previous sections described how to extend the EM algorithm to incorporate a Bayesian prior, the choice of seed clusters to initialize this procedure is also a key issue, as the EM algorithm is highly susceptible to finding local optima near its initial values. Assuming again that the clusters from interval $k-1$ provide a good starting point, an obvious seeding strategy is to assign the current data points to whichever prior cluster is closest. For this we use the (squared) Mahalanobis distance between the ith data point in sampling interval k and the jth cluster center estimated from interval $k-1$:

$$d_j^2(y_i) = (y_i - \hat{\mu}_j^{k-1})^T (\hat{\Sigma}_j^{k-1})^{-1} (y_i - \hat{\mu}_j^{k-1}) \,. \qquad (7)$$

Recall, however, that the EM algorithm is applied to a range of candidate model classes, with varying model order (numbers of clusters). The primary complication arises in cases where the candidate model order G_m is different from the model order estimated in the previous interval, $\hat{G}^{k-1}$. Such differences can naturally arise, for example, when neurons go silent or new neural signals are introduced between sampling intervals. Below we outline our approach for each of the three cases of varying G_m in relation to $\hat{G}^{k-1}$.

1) Case $G_m = \hat{G}^{k-1}$: The seed assignment process assigns each observation to the closest prior cluster; that is, for each i, the observation is assigned to the jth cluster, where j is the index that minimizes $d_j^2(y_i)$ in (7).

2) Case $G_m < \hat{G}^{k-1}$: The goal here is to produce a good clustering seed when $\Delta G = \hat{G}^{k-1} - G_m$ neuron(s) disappear (or perhaps become indistinguishable in the current feature space) between sampling intervals. To produce appropriate seeds, all $\binom{\hat{G}^{k-1}}{G_m}$ combinations of the $\hat{G}^{k-1}$ prior clusters are evaluated to determine which set of G_m clusters minimizes the sum of the squared Mahalanobis distance from the nearest prior cluster. This process tests the elimination of each possible prior cluster(s), keeping the best to inform the current data set.

3) Case $G_m > \hat{G}^{k-1}$: In this case, $\Delta G = G_m - \hat{G}^{k-1}$ "extra" seed clusters must be generated. Such a situation most commonly occurs when ΔG new neurons have been detected and a new cluster must be created for each. Another possibility is that the prior interval's clustering result was incorrect — with multiple neurons being inaccurately grouped into one cluster, and the current clustering step must rectify this.

The data points obtained in the current sampling interval are first assigned to the $\hat{G}^{k-1}$ prior clusters, as in the first case above, after which we wish to divide the cluster that is most likely to contain multiple neurons. Since such a group is likely to have a larger spread of points, the group with the largest average point-to-centroid Euclidean distance is chosen. A one-step divisive hierarchical clustering technique is applied to spilt the cluster. The above identification and splitting of groups is repeated as necessary for $\Delta G > 1$.

E. Selecting the Model Class $\mathcal{M}_m$

To start the model selection step, a Bayesian approach is used again, with the model probability:

$$P\big(\mathcal{M}_m \; Y^{1:k}\big) = \frac{p\big(Y^k \; Y^{1:k-1}, \mathcal{M}_m\big) P\big(\mathcal{M}_m \; Y^{1:k-1}\big)}{p\big(Y^k \; Y^{1:k-1}\big)} \,. \qquad (8)$$

This probability is difficult to compute because the evidence term $p\big(Y^k \; Y^{1:k-1}, \mathcal{M}_m\big)$ theoretically requires an integration over all possible parameters.

However, by employing Laplace's asymptotic approximation [13], [17], we can estimate a value of the evidence term while evaluating only at the MAP parameters $\hat{\Theta}_m^k$:

$$p\big(Y^k \; Y^{1:k-1}, \mathcal{M}_m\big) \approx p\big(Y^k \; \hat{\Theta}_m^k, \mathcal{M}_m\big) \cdot p\big(\hat{\Theta}_m^k \; \hat{\Theta}_m^{k-1}\big) (2\pi)^{n_m/2} \, \mathbf{H}_m(\hat{\Theta}_m^k)^{\,-1/2} \,, \qquad (9)$$

where n_m is the number of independent parameters in model $\mathcal{M}_m$. The first factor is the likelihood of the Gaussian mixture and the other factors, collectively known as the Ockham factor, as they penalize the complexity of the model parameterization, include the parameter prior (3) and the Hessian matrix $\mathbf{H}_m$, which we have determined analytically for the model classes under consideration. Most popular approaches to model selection, such as the Akaike Information Criterion (AIC) and Bayes Information Criterion (BIC), are essentially approximations to (9) and specific to the maximum likelihood case [17]. For our application, the Laplace approach naturally incorporates the prior on Θ_m^k.

The model class prior $P\big(\mathcal{M}_m \; Y^{1:k-1}\big)$ in (8) is simply the output of the previous clustering interval, under the assumption that the model class is constant. However, there exists some probability that the model class can change (e.g., new neural signal sources appear, or existing signal sources disappear). Thus, we use a weighted mixture of the previous result and a uniform prior:

$$P\big(\mathcal{M}_m \; Y^{1:k-1}\big) \qquad \alpha \, P\big(\mathcal{M}_m \; Y^{1:k-1}\big) + (1-\alpha)\frac{1}{\bar{M}} \;\; ,$$

where $\bar{M}$ is the total number of model classes under consideration. This in effect places a "forgetting factor" on the prior, governed by the parameter α (we use $\alpha = 0.95$), and ensures a nontrivial probability of each model class at every sampling interval.

IV. Tracking Clusters Across Time

Ultimately our goal is to "track" individual neurons across the recording session — that is, to associate specific neurons

with specific signal clusters over time. Viewing this as a data association task on the means, the quantity $\hat{\zeta}_{gj}^k$ already encodes the probability that current cluster $\mathcal{C}_g^k$ is associated with prior cluster $\mathcal{C}_j^{k-1}$, *relative* to all $\hat{G}^{k-1}+1$ components in the prior (3). Each current cluster $\mathcal{C}_g^k$ is therefore matched to its best prior cluster $\mathcal{C}_{j^*}^{k-1}$ via $j^* = \arg\max_j \hat{\zeta}_{gj}^k$ to track neurons over consecutive intervals. Thus, at the completion of the EM iterations, in addition to the model parameters $\hat{\Theta}_m^k$ and the cluster memberships $\hat{z}_{ig}^k$, the algorithm also yields the cluster associations $\hat{\zeta}_{gj}^k$ for tracking.

If j^* indicates a match to the uniform distribution, $\mathcal{C}_g^k$ is considered a new neuron, highlighting the importance of this uniform component of our prior. Disappearing neurons are identified during this process as well, simply by not matching prior clusters to any current clusters. Additionally, we do not restrict *multiple* current clusters $\mathcal{C}_g^k$ from matching to the same prior cluster $\mathcal{C}_j^{k-1}$, as we wish to mark these events as "splits" of the neuronal signals (when the signals of two (or more) neurons were previously indistinguishable and now separated).

V. Experimental Results

The proposed algorithm was applied to recordings from macaque parietal cortex, collected in acute recording sessions with platinum-iridium, 1.5 MΩ-impedance electrodes in a microdrive controlled by our autonomous positioning algorithm [2]–[4]. Spikes were detected from the recorded voltage stream according to a wavelet matching approach [18], aligned by their minimum, and projected onto a two-dimensional PCA feature space prior to classification.

As noted earlier, EM optimization of a Gaussian mixture model with ML parameters has shown its effectiveness in many clustering applications [13] and specifically spike sorting [1], [11], [12]. Thus, we compare our results to such a technique, which we have used for the past two years in hundreds of recording sessions. We originally chose this method based on its high rate of success compared to other spike sorting options, especially as our motivating applications requires sufficiently low computational cost for real-time usage and good results even in the case of small amounts of data.

In the implementation of the ML approach, seed clusters are generated from a standard hierarchical agglomerative technique and model order is selected according to Bayesian information criterion (BIC)[2], following the suggestions of [19]. Both the MAP and ML implementations use common-volume parsimonious models of the components' covariance matrices and the same uniform "background" mixture component to capture outliers. Below, we first examine sequences of consecutive sampling intervals in detail and then provide views of algorithm performance over longer time frames.

Fig. 2 displays clustering results over a sequence of six consecutive sampling intervals (or *steps*), chosen to highlight how the MAP algorithm enables neuron tracking over time, especially as compared to alternatives. Each sampling interval contains 10 seconds of data, with separating intervals of approximately 25 seconds during which no signals are sampled. For consistent visualization, we use the same PCA feature space for each step, rather than the PCA features of the individual steps (in which clustering took place). Although it is impossible to know the actual spike-neuron associations conclusively, the results are compared to a best-effort "ground truth" clustering of the data, as determined by a thorough manual examination of *both* the spikes' waveforms and PCA features in each sampling interval (whereas the automated clustering uses only PCA features). In addition to the MAP and ML algorithm results, a k-means clustering is also presented, with the number of clusters k manually selected to match the number of clusters in the ground truth results. Also listed for each step in Fig. 2 is the percentage of spikes correctly classified, calculated as follows: Each cluster is matched to the "truth" cluster sharing the most spikes, and the number of spikes these clusters have in common is considered correctly classified.

TABLE I
Cluster statistics of selected steps from sequence of Fig. 2.

Step	MAP					ML				
	g	Tr.	N_g	Err.	ΔFR	g	Tr.	N_g	Err.	ΔFR
1	1	A	49	0%	0%	1	A	48	2%	2%
	2	B	12	0%	0%	2	B	13	8%	8%
	3	C	17	0%	0%	3	C	17	0%	0%
2	1	A	73	0%	0%	1	A	76	4%	4%
	2	B	53	0%	0%	2	—	5	n/a	n/a
	3	C	65	0%	0%	3	C	115	86%	77%
3	1	A	73	4%	1%	1	A	74	3%	3%
	2	B	32	23%	23%	2	B	32	23%	23%
	3	C	36	17%	12%	3	C	36	17%	12%
						4	—	4	n/a	n/a
6	1	A	74	4%	4%	1	A	104	35%	35%
	2	B	29	7%	7%	2	—	1	n/a	n/a
	3	C	48	7%	7%	3	C	53	18%	18%
						4	—	3	n/a	n/a

Table I provides a detailed view of steps where the MAP and ML differed significantly. For these steps, Table I lists a) Tr., the "truth" cluster to which the gth cluster was matched; b) N_g, the number of spikes in gth cluster; c) Err., the percentage of falsely classified spikes for this cluster; d) ΔFR, the difference in firing rate between the cluster and its matching truth cluster. For individual cluster purposes, we defined the error as Err. $= \frac{\text{MC+FP}}{N_{g,\text{truth}}}$, where MC is the number of missed classifications and FP is the number of false positives.

Letters in Fig. 2 label "neuron identities," indicating the tracking of neurons across steps. Overall, notice that the MAP algorithm consistently identifies three clusters in approximately the same PCA position and maintains the neuron identities through all steps, matching the desired results. The ML algorithm often provides good results as well, but some steps (e.g., 2 and 6) show incongruous (though statistically sound) results, seemingly more volatile to noise variations. Meanwhile, the k-means solution is unreliable, even with the advantage of knowing the model order *a priori*.

The steps with poor results significantly impact our ability

[2] $BIC \equiv 2l_M(\hat{\Theta}_m^k \quad Y^k, \mathcal{M}_m) - n_m \log N$, for maximized mixture log-likelihood l_M, and number of independent model parameters n_m.

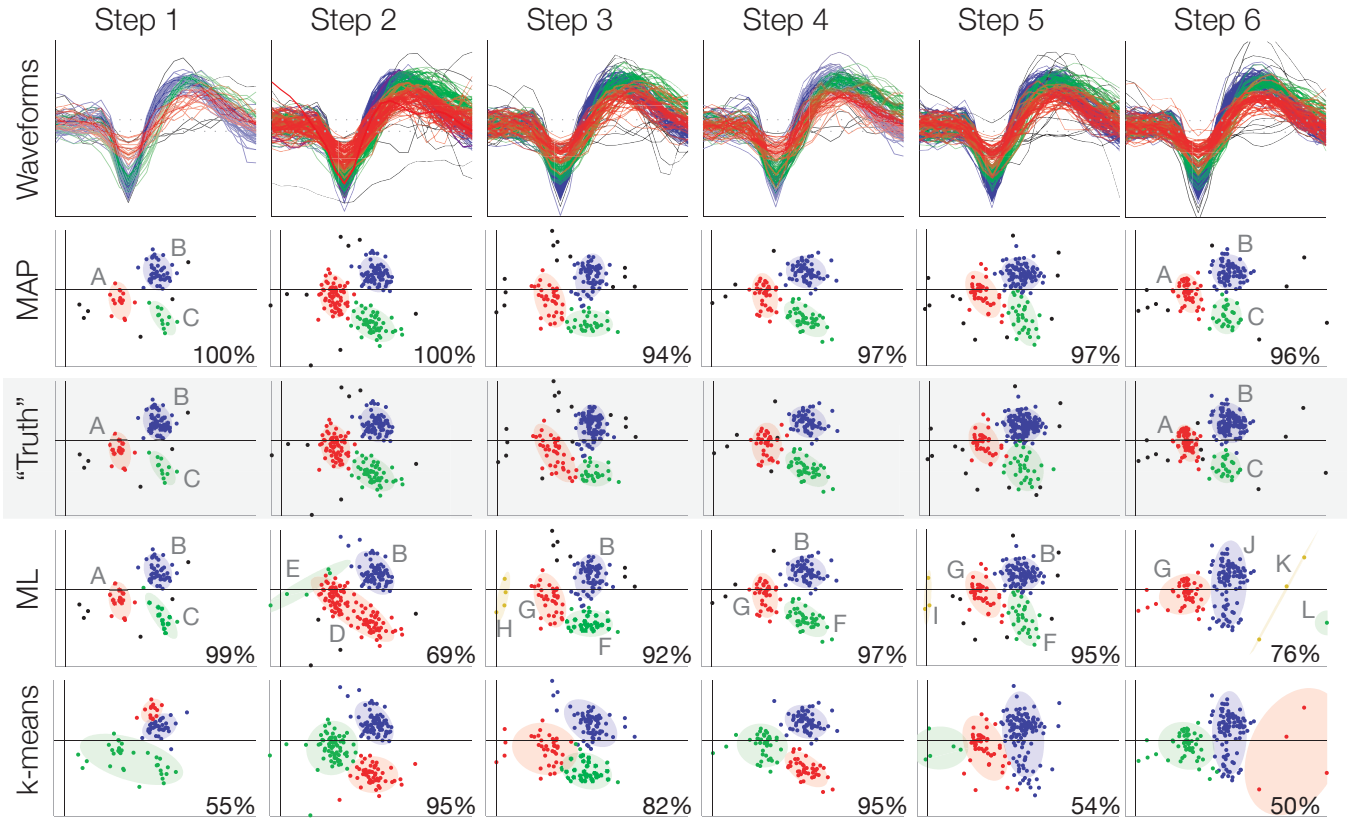

Fig. 2. Cluster results over six consecutive time steps in a common PCA space. Rows: (1) Extracted, aligned waveforms; (2) Results of our MAP algorithm; (3) Manual attempt at "ground truth"; (4) Baseline (ML) algorithm. (5) k-means, with $k = 3$. Shaded ellipses indicate $\sigma = 2$ for each cluster; percentile is of spikes classified correctly; capital letters label neuron identity (omitted in first two rows steps 2-5 since same throughout); black points are outliers.

to track neurons over consecutive steps, making the cost of misclassification on a single step high. For example, on step 2, the ML method groups most spikes from neurons A and C into a single cluster. When attempting to associate the clusters across time, this results in "losing" the two neurons, with the cluster interpreted as new (labeled D), rather than just tracking the neurons from step 1 as the MAP method does. Then, when the spikes from A and C are (mostly) correctly classified in step 3, they are actually considered as "new" neurons again (F and G) since they did not appear in the same position as in the previous step. This type of error is an example of misclassification preventing neuron tracking.

The consistency of the clustering outcome is a primary benefit of the proposed algorithm, as evidenced in Fig. 2. Although it is difficult to compellingly quantify this advantage, one metric to examine is the change in the number of clusters from step to step. Taking $\Psi = \sum_{s=1}^{S} \sum_{k=2}^{K} \hat{G}^k - \hat{G}^{k-1}$ over all time steps k of each recording session s for which we applied the proposed algorithm, we get a quantitative measure of "inconsistency" — note, however, that many changes of G are correct as the data change over a recording. Examining a set of $S = 100$ consecutive recording sessions, comprising about one month of recording trials and 21 914 total time steps k, $\Psi = 3516$ for the MAP algorithm, compared to $\Psi = 17\,646$ for the ML algorithm, an 80% decrease.

An example plot of the number of returned clusters $\hat{G}$ over an entire trial are shown in Fig. 3[3]. Clearly, the MAP algorithm provides a much more consistent model, though some spurious changes in the number of clusters are still evident. Several durations that neurons were tracked are shown as well as some related event types. Often, the regions where $\hat{G}$ varies more, as well as where neuron tracks change, correspond to motion of the electrode, as the control algorithm attempts to isolate a neuron. Where $\hat{G}$ changes for just a single step sometimes indicates an intermittent (temporarily inactive) neuron, and some are mistakes by our algorithm. These errors may have a variety of causes but are usually related to a lack of separability in PCA space or coincidental alignments of the spike detector's false positives (outliers).

VI. Discussion

The results presented above show how our MAP algorithm provides more consistent clustering, which enables tracking neurons from step to step. Additionally, it decreases the corruption of neuronal statistics, such as firing rate, caused by misclassification. Although we have focused on providing more consistent results, our algorithm has also performed well where the prior is not similar to the current clusters.

[3] $\hat{G} = 0$ occurs when no spikes are detected on the recording interval.

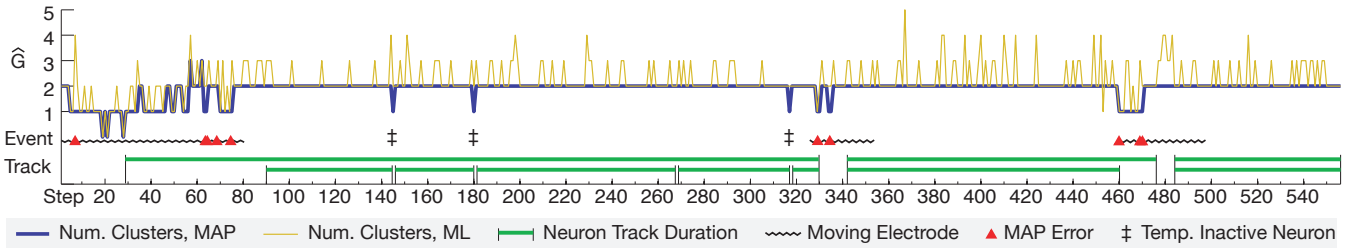

Fig. 3. Number of clusters $\hat{G}$ over time for four entire recording sessions, comparing consistency of MAP and ML clustering algorithms. The duration of several selected neuron tracks is marked, as well as several "event" types that may provide insight into why tracking failed during some steps.

The prior's construction as a mixture of densities effectively influences the cluster locations but assumes neither a certain number of clusters nor the *a priori* association of *particular* current and prior clusters. This is a key consideration for our prior, ensuring the algorithm is not unduly biased by the prior when evidence suggests the appearance (or disappearance) or neurons.

A few elements may be considered for future work. Most importantly, some clustering mistakes (particularly in model class selection) and temporarily inactive neurons are inevitable. We may seek to make the tracking algorithm more robust by incorporating prior information from several time steps and perhaps implementing a multiple hypothesis tracking approach. Also, choice of feature space, as well as a neuron's "dynamics" in this space, may be considered further.

In conclusion, we have detailed a Bayesian clustering algorithm to optimize a mixture model via EM. In addition to constructing an appropriate prior on cluster locations and adjusting the traditional EM approach to incorporate this term, we have created a new process for generating seed clusters and have proposed a suitable model class selection method. As a whole, this technique enables us to associate clusters across consecutive time intervals, and thus track neurons whose signals persist over many adjacent recording intervals. From an electrophysiology perspective, using a neuron tracking algorithm can increase the number of scientifically useful neurons identified on the signal and the reliability of the data. From the perspective of building an autonomous electrode positioning algorithm that tries to maximize the SNR of a particular neuron, consistently tracking the neurons' identities is essential to determining appropriate electrode control.

REFERENCES

[1] M. S. Lewicki, "A review of methods for spike sorting: the detection and classification of neural action potentials," *Network: Comput. Neural Syst.*, vol. 9, pp. R53–R78, 1998.

[2] J. G. Cham, E. A. Branchaud, Z. Nenadic, B. Greger, R. A. Andersen, and J. W. Burdick, "Semi-chronic motorized microdrive and control algorithm for autonomously isolating and maintaining optimal extracellular action potentials," *J. Neurophysiol.*, vol. 93, pp. 570–579, Jan. 2005.

[3] Z. Nenadic and J. W. Burdick, "A control algorithm for autonomous optimization of extracellular recordings," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 5, pp. 941–955, May 2006.

[4] E. A. Branchaud, "An algorithm for the autonomous isolation of neurons in extracellular recordings," Ph.D. dissertation, California Institute of Technology, 2006.

[5] J. G. Cham, M. T. Wolf, R. A. Andersen, and J. W. Burdick, "Miniature neural interface microdrive using parylene-coated layered manufacturing," in *IEEE/RAS-EMBS Intl. Conf. on Biomedical Robotics and Biomechatronics (BioRob)*, 2006.

[6] M. S. Fee, P. P. Mitra, and D. Kleinfeld, "Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-Gaussian variability," *J. Neurosci. Meth.*, vol. 69, no. 2, pp. 175–188, Nov. 1996.

[7] E. Hulata, R. Segev, and E. Ben-Jacob, "A method for spike sorting and detection based on wavelet packets and Shannon's mutual information," *J. Neurosci. Meth.*, vol. 117, no. 1, pp. 1–12, May 2002.

[8] F. Ohberg, H. Johansson, M. Bergenheim, J. Pedersen, and M. Djupsjobacka, "A neural network approach to real-time spike discrimination during simultaneous recording from several multi-unit nerve filaments," *J. Neurosci. Meth.*, vol. 64, no. 2, pp. 181–187, Feb. 1996.

[9] R. Q. Quiroga, Z. Nadasdy, and Y. Ben-Shaul, "Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering," *Neural Comput.*, vol. 16, no. 8, pp. 1661–1687, 2004.

[10] T. I. Aksenova, O. K. Chibirova, O. A. Dryga, I. V. Tetko, A.-L. Benabid, and A. E. P. Villa, "An unsupervised automatic method for sorting neuronal spike waveforms in awake and freely moving animals," *Methods*, vol. 30, no. 2, pp. 178–187, Jun. 2003.

[11] S. Shoham, M. R. Fellows, and R. A. Normann, "Robust, automatic spike sorting using mixtures of multivariate t-distributions," *J. Neurosci. Meth.*, vol. 127, no. 2, pp. 111–122, Aug. 2003.

[12] A. Bar-Hillel, A. Spiro, and E. Stark, "Spike sorting: Bayesian clustering of non-stationary data," *J. Neurosci. Meth.*, vol. 157, no. 2, pp. 303–316, Oct. 2006.

[13] G. McLachlan and D. Peel, *Finite Mixture Models*. Wiley Interscience, 2000.

[14] P. Cheeseman and J. Stutz, "Bayesian classification (AutoClass): Theory and results," in *Advances in Knowledge Discovery and Data Mining*, U. Fayyard, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds. Cambridge, MA: AAAI/MIT Press, 1996, ch. 6, pp. 61–83.

[15] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc. Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.

[16] G. Celeux and G. Govaert, "Gaussian pasimonious clustering models," *Pattern Recognition*, vol. 28, pp. 781–793, 1995.

[17] J. L. Beck and K.-V. Yuen, "Model selection using response measurements: Bayesian probabilistic approach," *J. Eng. Mechanics*, vol. 130, no. 2, pp. 192–203, 2004.

[18] Z. Nenadic and J. W. Burdick, "Spike detection using the continuous wavelet transform," *IEEE Trans. Biomed. Eng.*, vol. 52, no. 1, pp. 74–87, Jan. 2005.

[19] C. Fraley and A. E. Raftery, "How many clusters? Which clustering method? Answers via model-based cluster analysis," *Computer J.*, 1998.